


# Grafos

Ingeniería en Computación e Informática

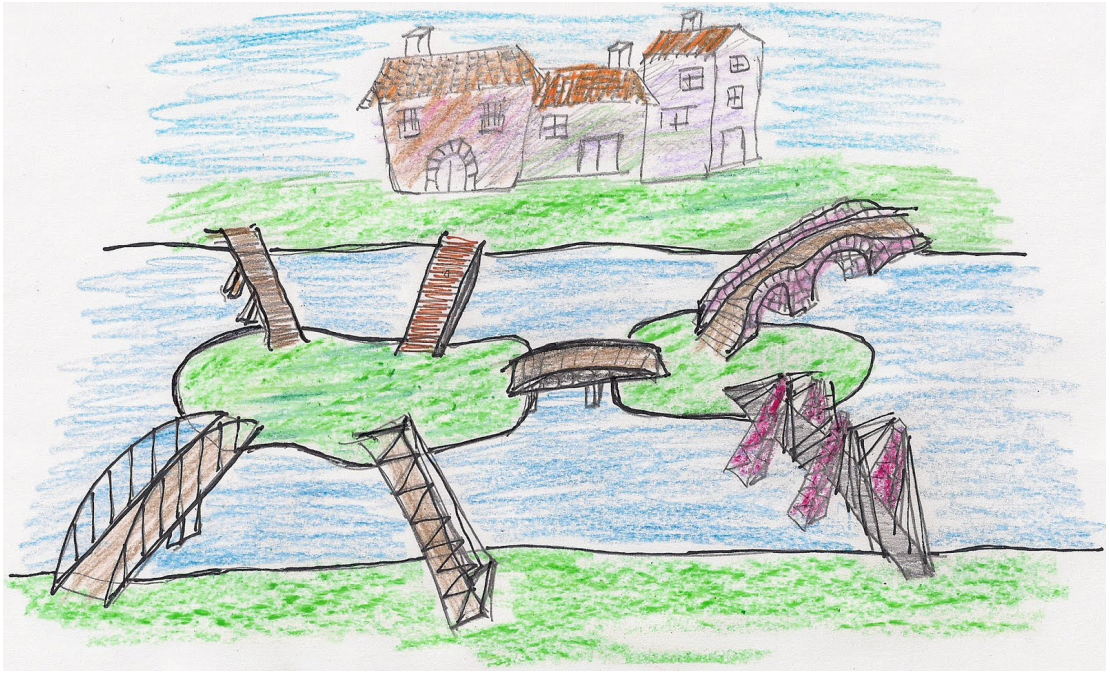
# Tabla de contenidos

# Introducción

- La primera publicación relacionada con la teoría de grafos se remonta a 1736 por Leonhard Euler al problema de los puentes de Königsberg.
- Varios resultados importantes se obtuvieron en el siglo XIX.
- Entre 1920 y 1930 se afianzó, extendió e intensificó el interés por la teoría de grafos.

# Introducción

El problema de los puentes de Königsberg consistía en encontrar un camino que recorriera los siete puentes del río Pregel en la ciudad de Königsberg, actualmente Kaliningrado, de modo que se recorrieran todos los puentes pasando una sola vez por cada uno de ellos.



# Introducción

- El primer texto apareció en 1936 escrito por König.
- El término *"grafo"*, proviene de la expresión *graphic notation* usada por primera vez por Frankland y posteriormente adoptada por Alexander Crum Brown en 1884, y hacía referencia a la representación gráfica de los enlaces entre los átomos de una molécula.
- Su interés se debe a su aplicabilidad en muchos campos como ciencias de la computación, química, investigación de operaciones, ingeniería eléctrica, lingüística y economía.

# Definición

- Los grafos son estructuras discretas que constan de vértices y aristas que conectan estos vértices.
- Hay diferentes tipos de grafos que difieren en la clase y número de aristas que conectan un par de vértices.

**Definición**

*Un grafo $G$ es un par ordenado $G = (V, E)$ formado por un conjunto finito de vértices $V$ y un conjunto $E$ de pares no ordenados de vértices distintos, es decir*

$$E \subset \{\{u, v\} \,|\, u, v \in V \wedge u \neq v\}$$

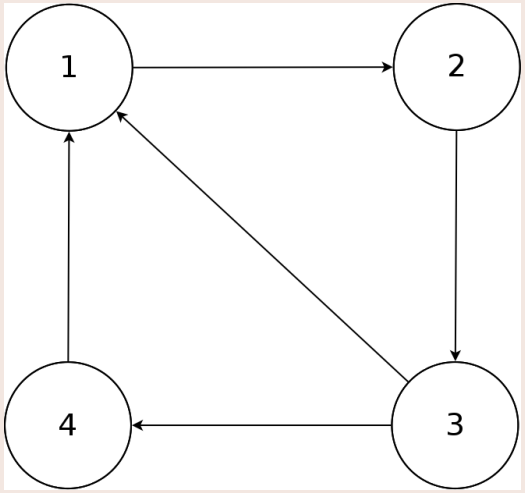
*A los elementos de $E$ se les denomina aristas o arcos.*

# Definición

## Grafo dirigido

### Grafo dirigido

Un grafo dirigido es un par ordenado $(V, E)$ donde V es un conjunto finito y $E \subset (V \times V) - \triangle$, siendo $\triangle = \{(x, x) : x \in V\}$.

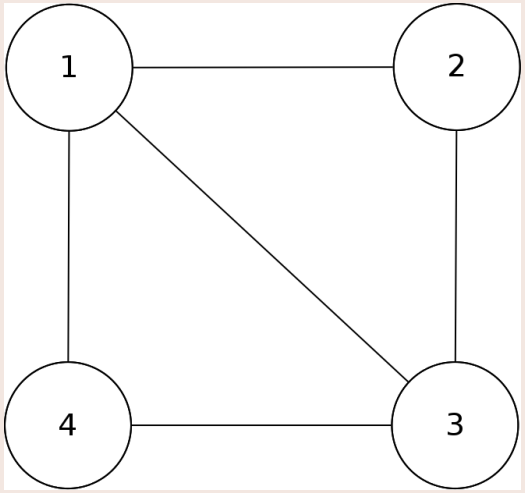


# Definición

## Grafo no dirigido

**Definición**

*Un grafo no dirigido es un par ordenado $(V, E)$ formado por un conjunto finito de vértices $V$ y una familia finita $E$ de aristas no orientadas $E = \{e_i\}_{i \in I}$ donde $I$ es un conjunto finito y $\forall i \in I$ se verifica que $e_i = \{u_i, v_i\}$ con $u_i, v_i \in V$.*



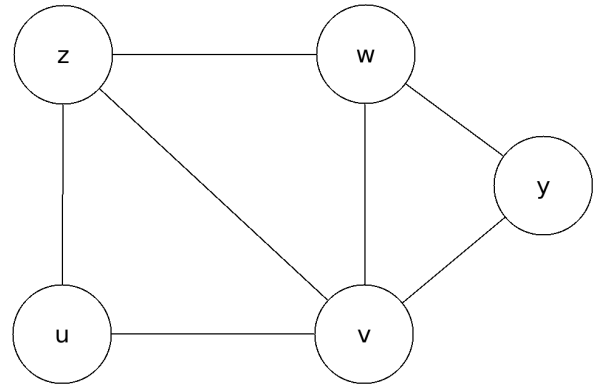

En un grafo no dirigido, dos aristas distintas pueden conectar los mismos vértices, esto es, que $E$ es una familia y no un conjunto, pudiendo tener elementos repetidos, en otras palabras se permite aristas del tipo $\{u, u\}$ denominadas *lazos* o *bucles*.

# Definición

## Orden y tamaño de un grafo

- El número de vértices de $G$ es $|V| = n$, el cual es el orden del grado del grafo $G$.
- El número de aristas $|E| = m$ es el tamaño del grafo $G$.
- Se representa un grafo mediante un dibujo donde los vértices son puntos y las aristas líneas que unen los vértices adyacentes.

El grafo de la figura es de orden 5 y tamaño 7, los vértices $u$ y $v$ son adyacentes, mientras que $u$ y $w$ son vértices independientes.



# Definición

## Grado de un grafo

**Definición**

*El grado de un vértice en un grafo no dirigido es el número de aristas incidentes con él, teniendo en cuenta que un lazo en un vértice contribuye dos veces al grado de ese vértice. Se denota el grado de un vértice $u$ por $deg(u)$.*

El máximo grado de un vértice en un grafo $G$ se denota por $\phi(G)$, mientras que el mínimo grado por $\Delta(G)$.

# Definición

## Densidad de un grafo

La densidad de un grafo $G$ se denota como $d(G)$, que se calcula de acuerdo a la ecuación 1.

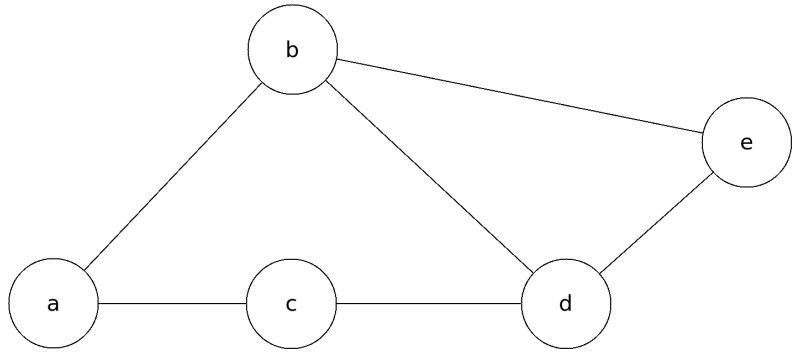
$$d(G) = \frac{2m}{n(n-1)} \tag{1}$$

Donde $0 < d(G) < 1$, si $d(G) = 0$ todos los vértices son aislados y si $d(G) = 1$ el grafo es completo. Si $d(G)$ es cercano a cero se dice que el grafo es disperso y, si $d(G)$ es cercano a 1 se dice que el grafo es denso.

# Definición

## Grafo conexo

Un grafo $G$ con al menos dos vértices, se dice que es conexo si cada par de vértices está conectado por una arista.




- $V = \{a, b, c, d, e\}$ y $E = \{(a, b), (a.c), (b, d), (b, e), (c, d), (d, e)\}$
- $n = 5$ y $m = 6$
- $\phi(G) = 3$ y $\delta(G) = 2$
- $d(G) = 0, 6$

# Definición

## Otros conceptos

- `Camino`: Un camino $P$ de longitud $n$ desde un vértice $v$ a un vértice $w$ se define como la secuencia de $n$ vértices que se debe conseguir para llegar del origen al vértice destino.

  $$P = \{v_1, \ldots, v_n\}$$

  De tal modo que: $v = v_1$; $v_1$ es adyacente a $v_{i+1}$, para $i = 1, 2, \ldots, n-1$ y $w = v_n$.

- `Camino cerrado`: El camino $P$ es cerrado si el primer y último vértice son iguales, es decir, si $v = w$.

- `Camino simple`: El camino es simple si todos sus vértices son distintos, con excepción del primero y el último, que pueden ser iguales.

# Definición

## Otros conceptos

- `Ciclo`: Un ciclo es un camino simple cerrado de longitud 3 o mayor. Un ciclo de longitud $k$ se llama $k$-ciclo.
- `Árbol`: Se dice que un grafo $G$ es del tipo árbol o árbol libre, si $G$ es un grafo conexo sin ciclos.
- `Grafo Completo`: Si cada vértice $v$ de $G$ es adyacente a todos los demás vértices de $G$. Un grafo completo de $n$ vértices tendrá $n(n-1)/2$ aristas.
- `Grafo Etiquetado`: Se dice que un grafo está etiquetado si sus aristas tienen asignado un valor. Si cada arista $a$ tiene un valor numérico no negativo $u(a)$, llamado peso o longitud de $a$, se dice que $G$ tiene peso. En este caso, cada camino $P$ de $G$ tendrá asociado un peso o longitud que será la suma de los pesos de las aristas que forman el camino $P$.

# Definición

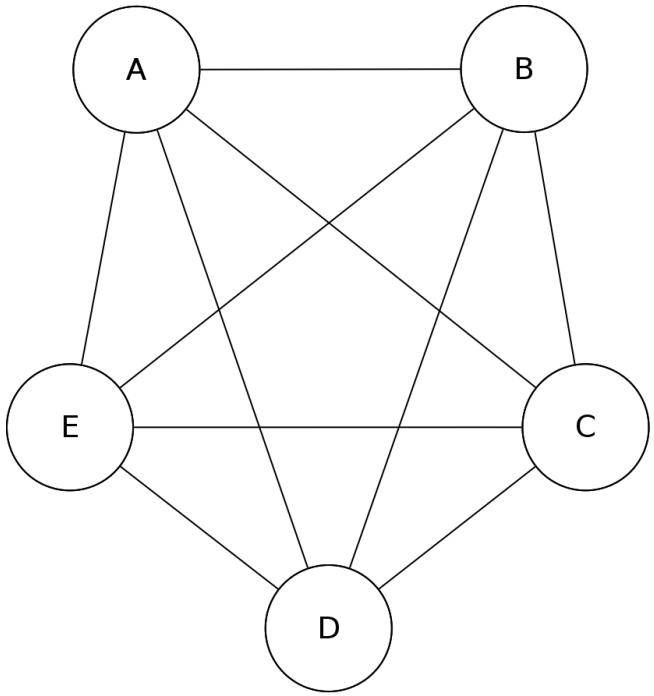
## Otros conceptos

- `Multigrafo`: Si al menos dos de sus vértices están conectados por dos aristas. En este caso, las aristas reciben el nombre de aristas múltiples o paralelas.

- `Subgrafo`: Dado un grafo $G = (V, E)$, $G' = (V', E')$ se denomina subgrafo de $G$ si: $V' \subseteq V$ y $E' \subseteq E$, donde cada arista de $E'$ es incidente con vértices de $V'$.

# Definición

## Otros conceptos




- Un camino $P$ para llegar del vértice $A$ al vértice $D$ puede ser: $A$-$B$-$C$-$D$ o $A$-$E$-$D$ o $A$-$D$ entre otros.
- El camino $A$-$C$-$D$-$A$ es un camino cerrado, el camino $A$-$C$-$D$ no lo es.
- El camino $A$-$C$-$D$-$A$ es un camino simple, el camino $A$-$C$-$B$-$D$-$C$ no lo es.
- El camino $A$-$C$-$D$-$A$ es un ciclo.
- El grafo es completo ya que todos los vértices se conectan con los demás.

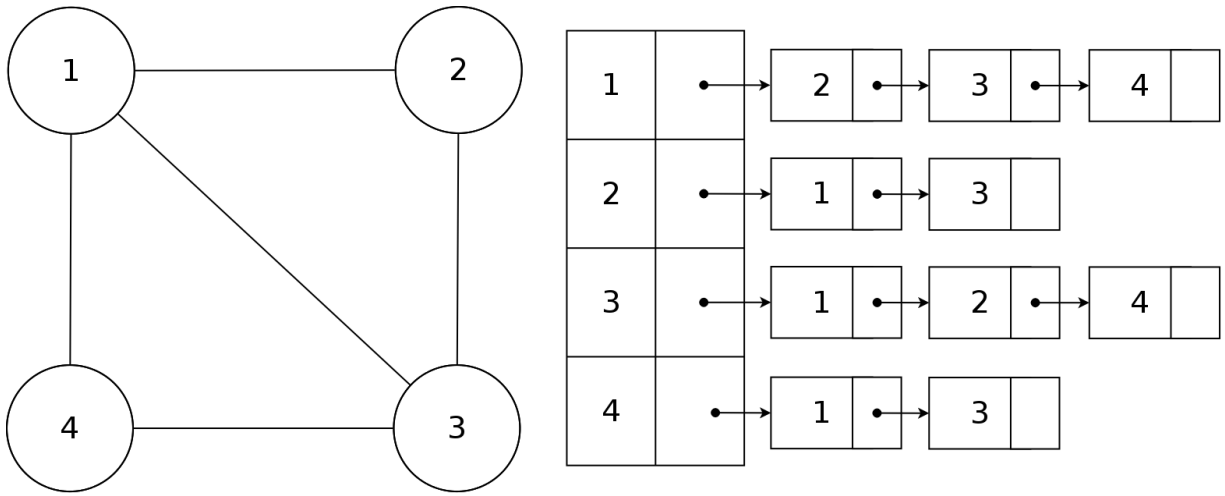
# Representación de Grafos

- Lista de adyacencia.
- Lista de Incidencia.
- Matriz de adyacencia.
- Matriz de Incidencia.

# Representación de Grafos

## Lista de adyacencia

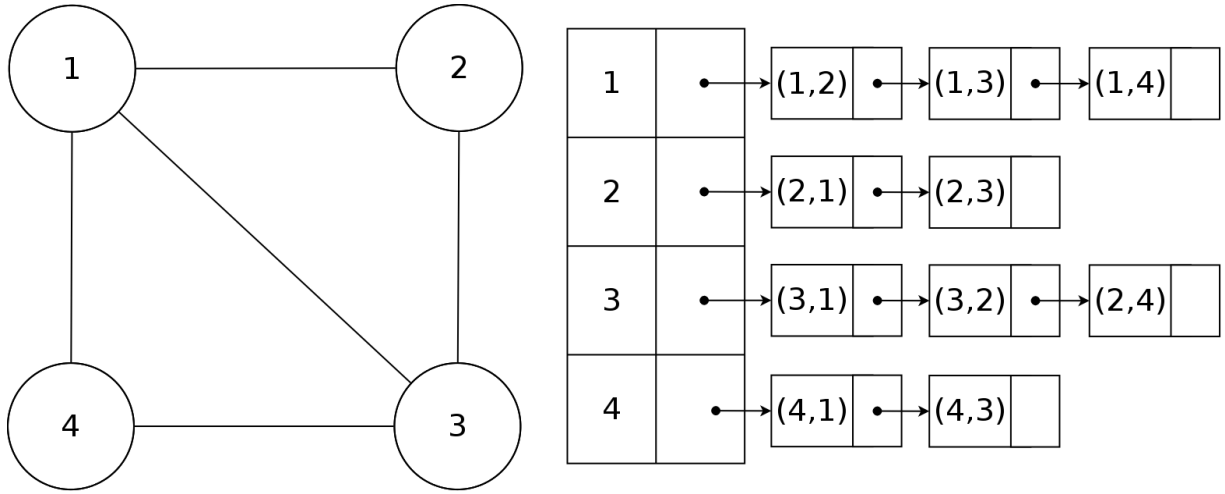
- Se genera una lista de los vértices.
- Cada elemento de la lista contiene una lista de los vértices adyacentes.
- Cuando es un grafo no dirigido, se deben colocar en ambos vértices sus adyacentes.



# Representación de Grafos

## Lista de incidencia

- Se genera una lista de los vértices.
- El grafo está representado por un arreglo de aristas, identificadas por un par de vértices, que son los que conecta esa arista.



# Representación de Grafos

## Matriz de adyacencia

- En la columna y en las filas se coloca el número del vértice correspondiente.
- En la intersección va 1 o verdadero si existe conexión o 0 o falso si no existe conexión. Si tiene peso la arista, puede ir el peso de ésta.
- Para grafos no dirigidos, se puede trabajar con la mitad de la matriz.

$$\begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}$$

# Representación de Grafos

## Matriz de incidencia

- Las columnas de la matriz representan las aristas del grafo.
- Las filas representan a los distintos vértices.
- Por cada vértice unido por una arista, se coloca un uno en el lugar correspondiente, y se llena el resto de las ubicaciones con ceros.

$$\begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \end{pmatrix}$$

# Representación de Grafos

## Complejidad de las operaciones

| | Lista Adyacencia | Lista de Incidencia | Matriz de Adyacencia | Matriz de Incidencia |
|---|---|---|---|---|
| Almacenamiento | $O(\|V\| + \|E\|)$ | $O(\|V\| + \|E\|)$ | $O(\|V\|^2)$ | $O(\|V\| \cdot \|E\|)$ |
| Agregar Vértice | $O(1)$ | $O(1)$ | $O(\|V\|^2)$ | $O(\|V\| \cdot \|E\|)$ |
| Agregar Arista | $O(1)$ | $O(1)$ | $O(1)$ | $O(\|V\| \cdot \|E\|)$ |
| Remover Vértice | $O(\|E\|)$ | $O(\|E\|)$ | $O(\|V\|^2)$ | $O(\|V\| \cdot \|E\|)$ |
| Remover Arista | $O(\|E\|)$ | $O(\|E\|)$ | $O(1)$ | $O(\|V\| \cdot \|E\|)$ |
| Vértices Adyacentes | $O(\|V\|)$ | $O(\|E\|)$ | $O(1)$ | $O(\|E\|)$ |

# Recorrido de Grafos

Se parte de un vértice dado y se visitan los vértices del grafo de manera ordenada y sistemática, pasando de un vértice a otro a través de las aristas del grafo.

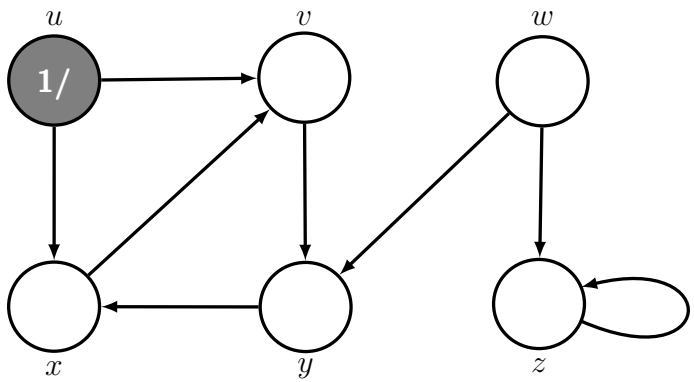
Tipos de recorridos:

- `Búsqueda primero en profundidad`: Equivalente al recorrido en preorden de un árbol.

- `Búsqueda primero en anchura`: Equivalente al recorrido de un árbol por niveles.

# Recorrido de Grafos

## Búsqueda primero en profundidad




**Algorithm** DFS

**Require:** $G$: Grafo.
1: **for** cada vértice $u \in G.V$ **do**
2: $u.color \leftarrow$ BLANCO
3: $u.\pi \leftarrow NULL$
4: **end for**
5: $tiempo \leftarrow 0$
6: **for** cada vértice $u \in G.V$ **do**
7: **if** $u.color =$ BLANCO **then**
8: DFS_VISITAR($G$, $u$)
9: **end if**
10: **end for**

$B$: regreso, $C$: cruce y $F$: arista revisada.

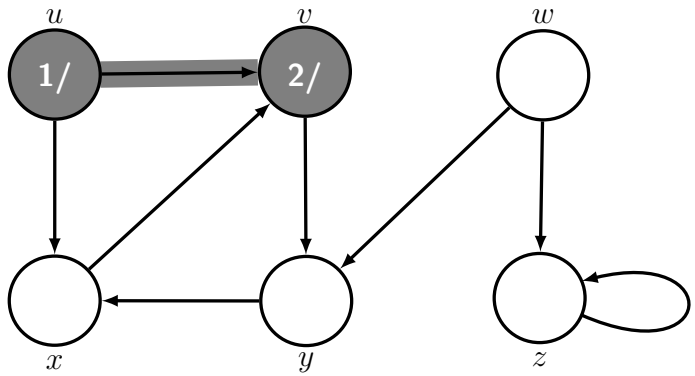
**Algorithm** DFS_VISITAR

**Require:** $G$: Grafo, $u$: vértice.
1: $tiempo \leftarrow tiempo + 1$
2: $u.d \leftarrow tiempo$
3: $u.color \leftarrow$ GRIS
4: **for** cada $v \in G.Adyacente(u)$ **do**
5: **if** $v.color =$ BLANCO **then**
6: $v.\pi \leftarrow u$
7: DFS_VISITAR($G$, $v$)
8: **end if**
9: **end for**
10: $u.color \leftarrow$ NEGRO
11: $tiempo \leftarrow tiempo + 1$
12: $u.f \leftarrow tiempo$

Las marcas de tiempo en los vértices indican el tiempo de descubrimiento / tiempos finales.

# Recorrido de Grafos

## Búsqueda primero en profundidad




**Algorithm** DFS

**Require:** $G$: Grafo.
1: **for** cada vértice $u \in G.V$ **do**
2: $u.color \leftarrow$ BLANCO
3: $u.\pi \leftarrow NULL$
4: **end for**
5: $tiempo \leftarrow 0$
6: **for** cada vértice $u \in G.V$ **do**
7: **if** $u.color =$ BLANCO **then**
8: DFS_VISITAR($G$, $u$)
9: **end if**
10: **end for**

$B$: regreso, $C$: cruce y $F$: arista revisada.

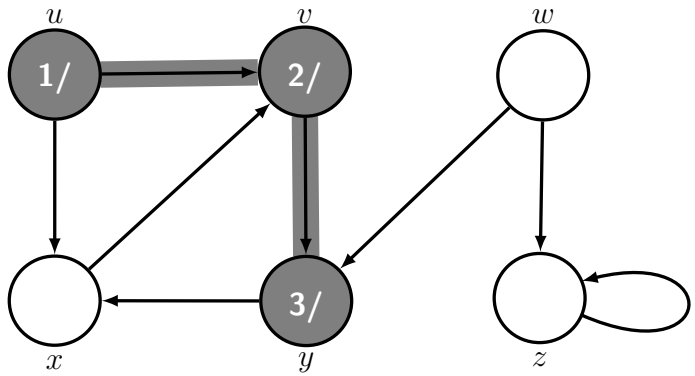
**Algorithm** DFS_VISITAR

**Require:** $G$: Grafo, $u$: vértice.
1: $tiempo \leftarrow tiempo + 1$
2: $u.d \leftarrow tiempo$
3: $u.color \leftarrow$ GRIS
4: **for** cada $v \in G.Adyacente(u)$ **do**
5: **if** $v.color =$ BLANCO **then**
6: $v.\pi \leftarrow u$
7: DFS_VISITAR($G$, $v$)
8: **end if**
9: **end for**
10: $u.color \leftarrow$ NEGRO
11: $tiempo \leftarrow tiempo + 1$
12: $u.f \leftarrow tiempo$

Las marcas de tiempo en los vértices indican el tiempo de descubrimiento / tiempos finales.

# Recorrido de Grafos

## Búsqueda primero en profundidad




**Algorithm** DFS

**Require:** $G$: Grafo.
1: **for** cada vértice $u \in G.V$ **do**
2: $u.color \leftarrow$ BLANCO
3: $u.\pi \leftarrow NULL$
4: **end for**
5: $tiempo \leftarrow 0$
6: **for** cada vértice $u \in G.V$ **do**
7: **if** $u.color =$ BLANCO **then**
8: DFS_VISITAR($G$, $u$)
9: **end if**
10: **end for**

$B$: regreso, $C$: cruce y $F$: arista revisada.

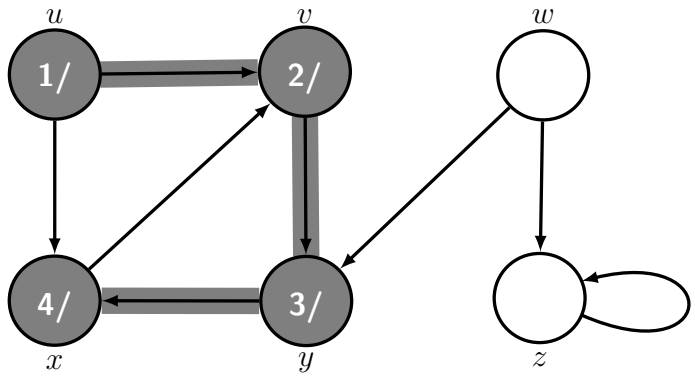
**Algorithm** DFS_VISITAR

**Require:** $G$: Grafo, $u$: vértice.
1: $tiempo \leftarrow tiempo + 1$
2: $u.d \leftarrow tiempo$
3: $u.color \leftarrow$ GRIS
4: **for** cada $v \in G.Adyacente(u)$ **do**
5: **if** $v.color =$ BLANCO **then**
6: $v.\pi \leftarrow u$
7: DFS_VISITAR($G$, $v$)
8: **end if**
9: **end for**
10: $u.color \leftarrow$ NEGRO
11: $tiempo \leftarrow tiempo + 1$
12: $u.f \leftarrow tiempo$

Las marcas de tiempo en los vértices indican el tiempo de descubrimiento / tiempos finales.

# Recorrido de Grafos

## Búsqueda primero en profundidad




**Algorithm** DFS

**Require:** $G$: Grafo.
1: **for** cada vértice $u \in G.V$ **do**
2: $u.color \leftarrow$ BLANCO
3: $u.\pi \leftarrow NULL$
4: **end for**
5: $tiempo \leftarrow 0$
6: **for** cada vértice $u \in G.V$ **do**
7: **if** $u.color =$ BLANCO **then**
8: DFS_VISITAR($G$, $u$)
9: **end if**
10: **end for**

$B$: regreso, $C$: cruce y $F$: arista revisada.

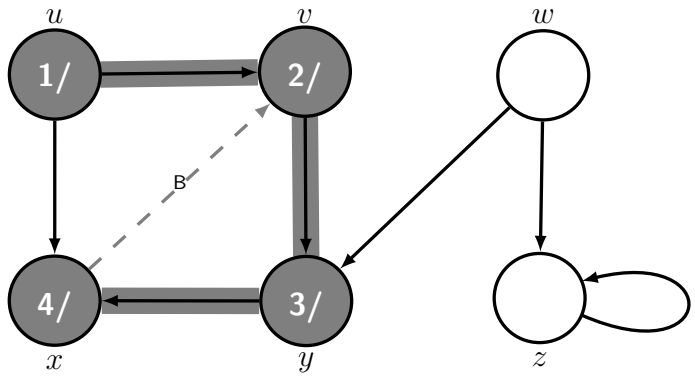
**Algorithm** DFS_VISITAR

**Require:** $G$: Grafo, $u$: vértice.
1: $tiempo \leftarrow tiempo + 1$
2: $u.d \leftarrow tiempo$
3: $u.color \leftarrow$ GRIS
4: **for** cada $v \in G.Adyacente(u)$ **do**
5: **if** $v.color =$ BLANCO **then**
6: $v.\pi \leftarrow u$
7: DFS_VISITAR($G$, $v$)
8: **end if**
9: **end for**
10: $u.color \leftarrow$ NEGRO
11: $tiempo \leftarrow tiempo + 1$
12: $u.f \leftarrow tiempo$

Las marcas de tiempo en los vértices indican el tiempo de descubrimiento / tiempos finales.

# Recorrido de Grafos

## Búsqueda primero en profundidad




**Algorithm** DFS

**Require:** $G$: Grafo.
1: **for** cada vértice $u \in G.V$ **do**
2: $u.color \leftarrow$ BLANCO
3: $u.\pi \leftarrow NULL$
4: **end for**
5: $tiempo \leftarrow 0$
6: **for** cada vértice $u \in G.V$ **do**
7: **if** $u.color =$ BLANCO **then**
8: DFS_VISITAR($G$, $u$)
9: **end if**
10: **end for**

$B$: regreso, $C$: cruce y $F$: arista revisada.

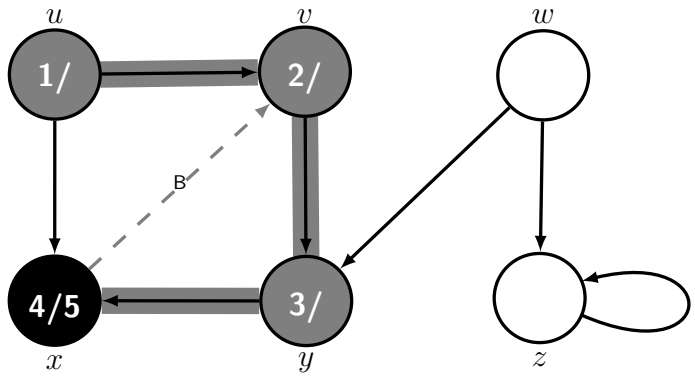
**Algorithm** DFS_VISITAR

**Require:** $G$: Grafo, $u$: vértice.
1: $tiempo \leftarrow tiempo + 1$
2: $u.d \leftarrow tiempo$
3: $u.color \leftarrow$ GRIS
4: **for** cada $v \in G.Adyacente(u)$ **do**
5: **if** $v.color =$ BLANCO **then**
6: $v.\pi \leftarrow u$
7: DFS_VISITAR($G$, $v$)
8: **end if**
9: **end for**
10: $u.color \leftarrow$ NEGRO
11: $tiempo \leftarrow tiempo + 1$
12: $u.f \leftarrow tiempo$

Las marcas de tiempo en los vértices indican el tiempo de descubrimiento / tiempos finales.

# Recorrido de Grafos

## Búsqueda primero en profundidad




**Algorithm** DFS

**Require:** $G$: Grafo.
1: **for** cada vértice $u \in G.V$ **do**
2: $u.color \leftarrow$ BLANCO
3: $u.\pi \leftarrow NULL$
4: **end for**
5: $tiempo \leftarrow 0$
6: **for** cada vértice $u \in G.V$ **do**
7: **if** $u.color =$ BLANCO **then**
8: DFS_VISITAR($G$, $u$)
9: **end if**
10: **end for**

$B$: regreso, $C$: cruce y $F$: arista revisada.

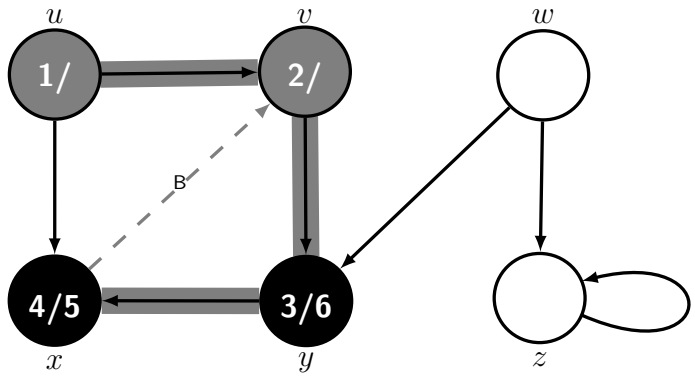
**Algorithm** DFS_VISITAR

**Require:** $G$: Grafo, $u$: vértice.
1: $tiempo \leftarrow tiempo + 1$
2: $u.d \leftarrow tiempo$
3: $u.color \leftarrow$ GRIS
4: **for** cada $v \in G.Adyacente(u)$ **do**
5: **if** $v.color =$ BLANCO **then**
6: $v.\pi \leftarrow u$
7: DFS_VISITAR($G$, $v$)
8: **end if**
9: **end for**
10: $u.color \leftarrow$ NEGRO
11: $tiempo \leftarrow tiempo + 1$
12: $u.f \leftarrow tiempo$

Las marcas de tiempo en los vértices indican el tiempo de descubrimiento / tiempos finales.

# Recorrido de Grafos

## Búsqueda primero en profundidad




**Algorithm** DFS

**Require:** $G$: Grafo.
1: **for** cada vértice $u \in G.V$ **do**
2: $u.color \leftarrow$ BLANCO
3: $u.\pi \leftarrow NULL$
4: **end for**
5: $tiempo \leftarrow 0$
6: **for** cada vértice $u \in G.V$ **do**
7: **if** $u.color =$ BLANCO **then**
8: DFS_VISITAR($G$, $u$)
9: **end if**
10: **end for**

$B$: regreso, $C$: cruce y $F$: arista revisada.

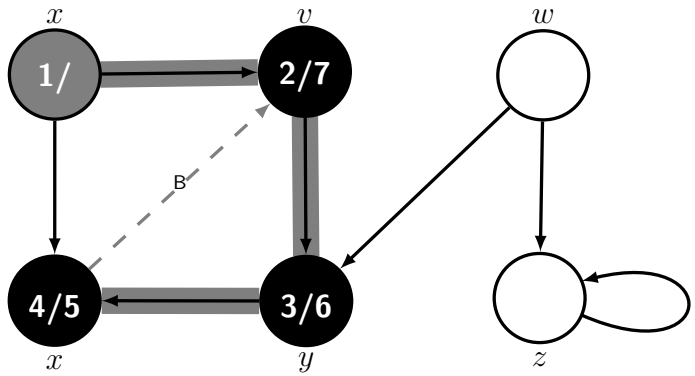
**Algorithm** DFS_VISITAR

**Require:** $G$: Grafo, $u$: vértice.
1: $tiempo \leftarrow tiempo + 1$
2: $u.d \leftarrow tiempo$
3: $u.color \leftarrow$ GRIS
4: **for** cada $v \in G.Adyacente(u)$ **do**
5: **if** $v.color =$ BLANCO **then**
6: $v.\pi \leftarrow u$
7: DFS_VISITAR($G$, $v$)
8: **end if**
9: **end for**
10: $u.color \leftarrow$ NEGRO
11: $tiempo \leftarrow tiempo + 1$
12: $u.f \leftarrow tiempo$

Las marcas de tiempo en los vértices indican el tiempo de descubrimiento / tiempos finales.

# Recorrido de Grafos

## Búsqueda primero en profundidad




**Algorithm** DFS

**Require:** $G$: Grafo.
1: **for** cada vértice $u \in G.V$ **do**
2: $u.color \leftarrow$ BLANCO
3: $u.\pi \leftarrow NULL$
4: **end for**
5: $tiempo \leftarrow 0$
6: **for** cada vértice $u \in G.V$ **do**
7: **if** $u.color =$ BLANCO **then**
8: DFS_VISITAR($G$, $u$)
9: **end if**
10: **end for**

$B$: regreso, $C$: cruce y $F$: arista revisada.

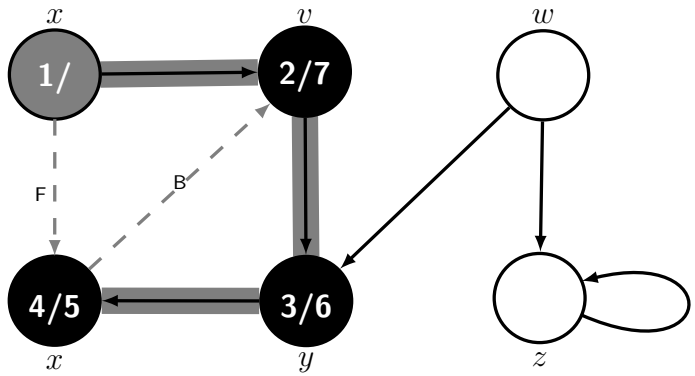
**Algorithm** DFS_VISITAR

**Require:** $G$: Grafo, $u$: vértice.
1: $tiempo \leftarrow tiempo + 1$
2: $u.d \leftarrow tiempo$
3: $u.color \leftarrow$ GRIS
4: **for** cada $v \in G.Adyacente(u)$ **do**
5: **if** $v.color =$ BLANCO **then**
6: $v.\pi \leftarrow u$
7: DFS_VISITAR($G$, $v$)
8: **end if**
9: **end for**
10: $u.color \leftarrow$ NEGRO
11: $tiempo \leftarrow tiempo + 1$
12: $u.f \leftarrow tiempo$

Las marcas de tiempo en los vértices indican el tiempo de descubrimiento / tiempos finales.

# Recorrido de Grafos

## Búsqueda primero en profundidad




**Algorithm** DFS

**Require:** $G$: Grafo.
1: **for** cada vértice $u \in G.V$ **do**
2: $u.color \leftarrow$ BLANCO
3: $u.\pi \leftarrow NULL$
4: **end for**
5: $tiempo \leftarrow 0$
6: **for** cada vértice $u \in G.V$ **do**
7: **if** $u.color =$ BLANCO **then**
8: DFS_VISITAR($G$, $u$)
9: **end if**
10: **end for**

$B$: regreso, $C$: cruce y $F$: arista revisada.

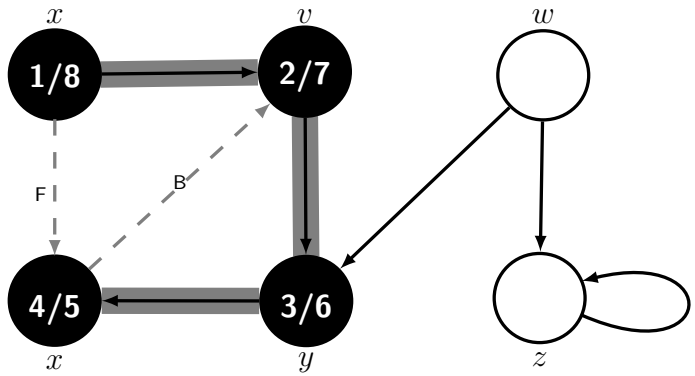
**Algorithm** DFS_VISITAR

**Require:** $G$: Grafo, $u$: vértice.
1: $tiempo \leftarrow tiempo + 1$
2: $u.d \leftarrow tiempo$
3: $u.color \leftarrow$ GRIS
4: **for** cada $v \in G.Adyacente(u)$ **do**
5: **if** $v.color =$ BLANCO **then**
6: $v.\pi \leftarrow u$
7: DFS_VISITAR($G$, $v$)
8: **end if**
9: **end for**
10: $u.color \leftarrow$ NEGRO
11: $tiempo \leftarrow tiempo + 1$
12: $u.f \leftarrow tiempo$

Las marcas de tiempo en los vértices indican el tiempo de descubrimiento / tiempos finales.

# Recorrido de Grafos

## Búsqueda primero en profundidad




**Algorithm** DFS

**Require:** $G$: Grafo.
1: **for** cada vértice $u \in G.V$ **do**
2: $u.color \leftarrow$ BLANCO
3: $u.\pi \leftarrow NULL$
4: **end for**
5: $tiempo \leftarrow 0$
6: **for** cada vértice $u \in G.V$ **do**
7: **if** $u.color =$ BLANCO **then**
8: DFS_VISITAR($G$, $u$)
9: **end if**
10: **end for**

$B$: regreso, $C$: cruce y $F$: arista revisada.

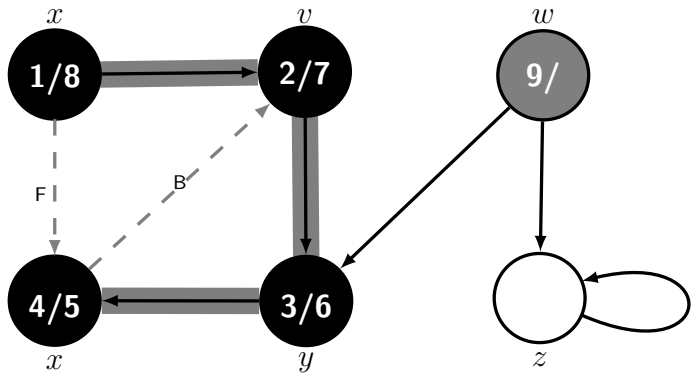
**Algorithm** DFS_VISITAR

**Require:** $G$: Grafo, $u$: vértice.
1: $tiempo \leftarrow tiempo + 1$
2: $u.d \leftarrow tiempo$
3: $u.color \leftarrow$ GRIS
4: **for** cada $v \in G.Adyacente(u)$ **do**
5: **if** $v.color =$ BLANCO **then**
6: $v.\pi \leftarrow u$
7: DFS_VISITAR($G$, $v$)
8: **end if**
9: **end for**
10: $u.color \leftarrow$ NEGRO
11: $tiempo \leftarrow tiempo + 1$
12: $u.f \leftarrow tiempo$

Las marcas de tiempo en los vértices indican el tiempo de descubrimiento / tiempos finales.

# Recorrido de Grafos

## Búsqueda primero en profundidad




**Algorithm** DFS

**Require:** $G$: Grafo.
1: **for** cada vértice $u \in G.V$ **do**
2: $u.color \leftarrow$ BLANCO
3: $u.\pi \leftarrow NULL$
4: **end for**
5: $tiempo \leftarrow 0$
6: **for** cada vértice $u \in G.V$ **do**
7: **if** $u.color =$ BLANCO **then**
8: DFS_VISITAR($G$, $u$)
9: **end if**
10: **end for**

$B$: regreso, $C$: cruce y $F$: arista revisada.

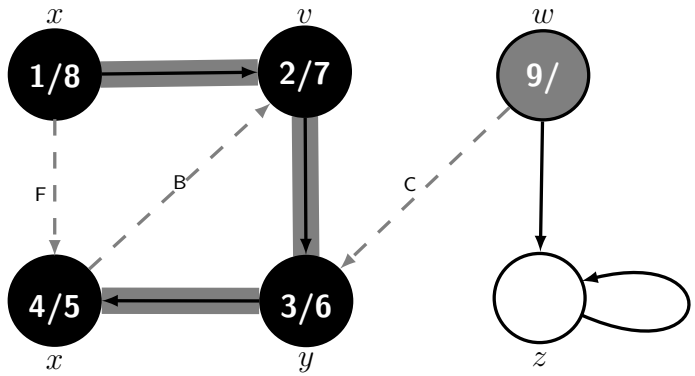
**Algorithm** DFS_VISITAR

**Require:** $G$: Grafo, $u$: vértice.
1: $tiempo \leftarrow tiempo + 1$
2: $u.d \leftarrow tiempo$
3: $u.color \leftarrow$ GRIS
4: **for** cada $v \in G.Adyacente(u)$ **do**
5: **if** $v.color =$ BLANCO **then**
6: $v.\pi \leftarrow u$
7: DFS_VISITAR($G$, $v$)
8: **end if**
9: **end for**
10: $u.color \leftarrow$ NEGRO
11: $tiempo \leftarrow tiempo + 1$
12: $u.f \leftarrow tiempo$

Las marcas de tiempo en los vértices indican el tiempo de descubrimiento / tiempos finales.

# Recorrido de Grafos

## Búsqueda primero en profundidad




**Algorithm** DFS

**Require:** $G$: Grafo.
1: **for** cada vértice $u \in G.V$ **do**
2: $u.color \leftarrow$ BLANCO
3: $u.\pi \leftarrow NULL$
4: **end for**
5: $tiempo \leftarrow 0$
6: **for** cada vértice $u \in G.V$ **do**
7: **if** $u.color =$ BLANCO **then**
8: DFS_VISITAR($G$, $u$)
9: **end if**
10: **end for**

$B$: regreso, $C$: cruce y $F$: arista revisada.

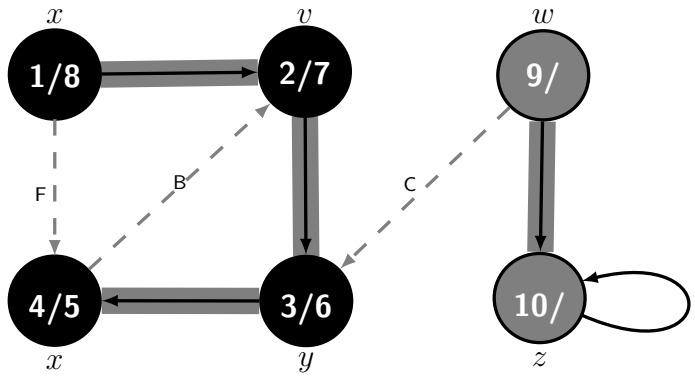
**Algorithm** DFS_VISITAR

**Require:** $G$: Grafo, $u$: vértice.
1: $tiempo \leftarrow tiempo + 1$
2: $u.d \leftarrow tiempo$
3: $u.color \leftarrow$ GRIS
4: **for** cada $v \in G.Adyacente(u)$ **do**
5: **if** $v.color =$ BLANCO **then**
6: $v.\pi \leftarrow u$
7: DFS_VISITAR($G$, $v$)
8: **end if**
9: **end for**
10: $u.color \leftarrow$ NEGRO
11: $tiempo \leftarrow tiempo + 1$
12: $u.f \leftarrow tiempo$

Las marcas de tiempo en los vértices indican el tiempo de descubrimiento / tiempos finales.

# Recorrido de Grafos

## Búsqueda primero en profundidad




**Algorithm** DFS

**Require:** $G$: Grafo.
1: **for** cada vértice $u \in G.V$ **do**
2: $u.color \leftarrow$ BLANCO
3: $u.\pi \leftarrow NULL$
4: **end for**
5: $tiempo \leftarrow 0$
6: **for** cada vértice $u \in G.V$ **do**
7: **if** $u.color =$ BLANCO **then**
8: DFS_VISITAR($G$, $u$)
9: **end if**
10: **end for**

$B$: regreso, $C$: cruce y $F$: arista revisada.

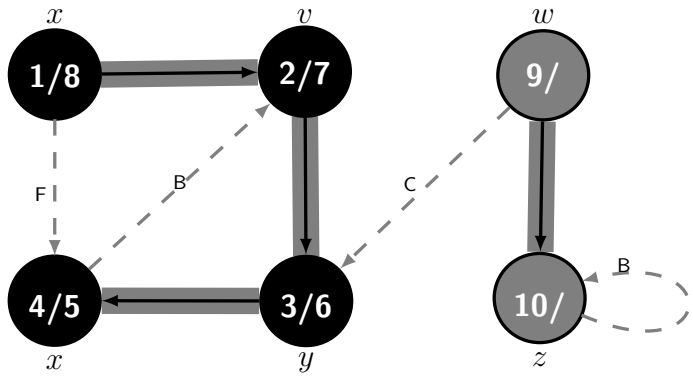
**Algorithm** DFS_VISITAR

**Require:** $G$: Grafo, $u$: vértice.
1: $tiempo \leftarrow tiempo + 1$
2: $u.d \leftarrow tiempo$
3: $u.color \leftarrow$ GRIS
4: **for** cada $v \in G.Adyacente(u)$ **do**
5: **if** $v.color =$ BLANCO **then**
6: $v.\pi \leftarrow u$
7: DFS_VISITAR($G$, $v$)
8: **end if**
9: **end for**
10: $u.color \leftarrow$ NEGRO
11: $tiempo \leftarrow tiempo + 1$
12: $u.f \leftarrow tiempo$

Las marcas de tiempo en los vértices indican el tiempo de descubrimiento / tiempos finales.

# Recorrido de Grafos

## Búsqueda primero en profundidad




**Algorithm** DFS

**Require:** $G$: Grafo.
1: **for** cada vértice $u \in G.V$ **do**
2: $u.color \leftarrow$ BLANCO
3: $u.\pi \leftarrow NULL$
4: **end for**
5: $tiempo \leftarrow 0$
6: **for** cada vértice $u \in G.V$ **do**
7: **if** $u.color =$ BLANCO **then**
8: DFS_VISITAR($G$, $u$)
9: **end if**
10: **end for**

$B$: regreso, $C$: cruce y $F$: arista revisada.

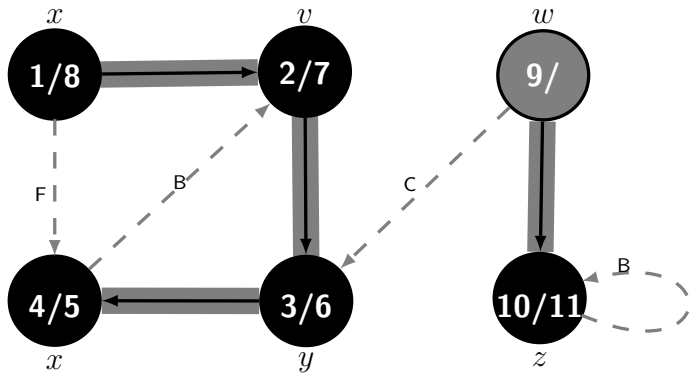
**Algorithm** DFS_VISITAR

**Require:** $G$: Grafo, $u$: vértice.
1: $tiempo \leftarrow tiempo + 1$
2: $u.d \leftarrow tiempo$
3: $u.color \leftarrow$ GRIS
4: **for** cada $v \in G.Adyacente(u)$ **do**
5: **if** $v.color =$ BLANCO **then**
6: $v.\pi \leftarrow u$
7: DFS_VISITAR($G$, $v$)
8: **end if**
9: **end for**
10: $u.color \leftarrow$ NEGRO
11: $tiempo \leftarrow tiempo + 1$
12: $u.f \leftarrow tiempo$

Las marcas de tiempo en los vértices indican el tiempo de descubrimiento / tiempos finales.

# Recorrido de Grafos

## Búsqueda primero en profundidad




**Algorithm** DFS

**Require:** $G$: Grafo.
1: **for** cada vértice $u \in G.V$ **do**
2: $u.color \leftarrow$ BLANCO
3: $u.\pi \leftarrow NULL$
4: **end for**
5: $tiempo \leftarrow 0$
6: **for** cada vértice $u \in G.V$ **do**
7: **if** $u.color =$ BLANCO **then**
8: DFS_VISITAR($G$, $u$)
9: **end if**
10: **end for**

$B$: regreso, $C$: cruce y $F$: arista revisada.

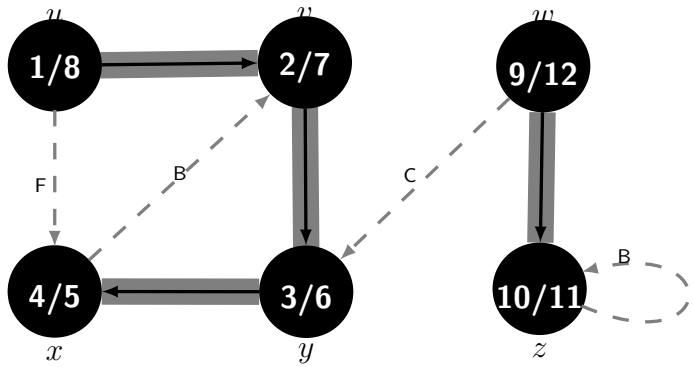
**Algorithm** DFS_VISITAR

**Require:** $G$: Grafo, $u$: vértice.
1: $tiempo \leftarrow tiempo + 1$
2: $u.d \leftarrow tiempo$
3: $u.color \leftarrow$ GRIS
4: **for** cada $v \in G.Adyacente(u)$ **do**
5: **if** $v.color =$ BLANCO **then**
6: $v.\pi \leftarrow u$
7: DFS_VISITAR($G$, $v$)
8: **end if**
9: **end for**
10: $u.color \leftarrow$ NEGRO
11: $tiempo \leftarrow tiempo + 1$
12: $u.f \leftarrow tiempo$

Las marcas de tiempo en los vértices indican el tiempo de descubrimiento / tiempos finales.

# Recorrido de Grafos

## Búsqueda primero en profundidad




**Algorithm** DFS

**Require:** $G$: Grafo.
1: **for** cada vértice $u \in G.V$ **do**
2: $u.color \leftarrow$ BLANCO
3: $u.\pi \leftarrow NULL$
4: **end for**
5: $tiempo \leftarrow 0$
6: **for** cada vértice $u \in G.V$ **do**
7: **if** $u.color =$ BLANCO **then**
8: DFS_VISITAR($G$, $u$)
9: **end if**
10: **end for**

$B$: regreso, $C$: cruce y $F$: arista revisada.

**Algorithm** DFS_VISITAR

**Require:** $G$: Grafo, $u$: vértice.
1: $tiempo \leftarrow tiempo + 1$
2: $u.d \leftarrow tiempo$
3: $u.color \leftarrow$ GRIS
4: **for** cada $v \in G.Adyacente(u)$ **do**
5: **if** $v.color =$ BLANCO **then**
6: $v.\pi \leftarrow u$
7: DFS_VISITAR($G$, $v$)
8: **end if**
9: **end for**
10: $u.color \leftarrow$ NEGRO
11: $tiempo \leftarrow tiempo + 1$
12: $u.f \leftarrow tiempo$

Las marcas de tiempo en los vértices indican el tiempo de descubrimiento / tiempos finales.

# Recorrido de Grafos

## Búsqueda primero en profundidad

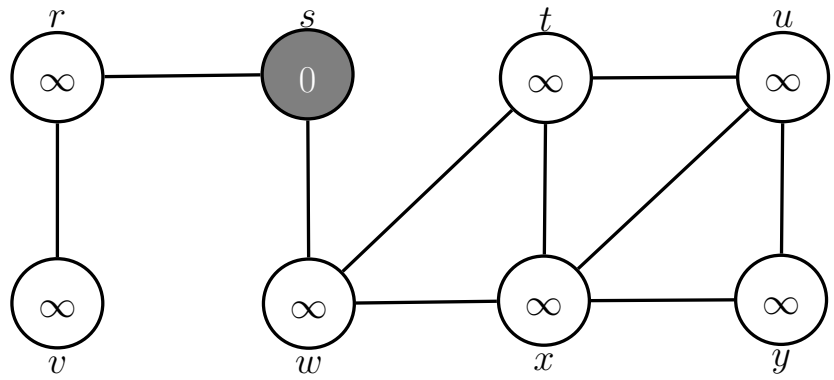
- Salida: $x$, $y$, $v$, $u$, $z$, $w$.
- El Eficiencia del algoritmo es $O(|V| + |E|)$.

# Recorrido de Grafos

## Búsqueda primero en anchura

**Algorithm** BFS

**Require:** $G$: Grafo, $s$: vértice inicial.
1: **for** cada vértice $u \in G.V - \{s\}$ **do**
2: $u.color \leftarrow$ BLANCO
3: $u.d \leftarrow \infty$
4: $u.\pi \leftarrow NULL$
5: **end for**
6: $s.color \leftarrow$ GRIS
7: $s.d \leftarrow 0$
8: $s.\pi \leftarrow NULL$
9: $Q \leftarrow \emptyset$
10: Encolar($Q$, $s$)
11: **while** $Q \neq \emptyset$ **do**
12: $u \leftarrow$ Desencolar($Q$)
13: **for** cada $v \in G.Adyacente(u)$ **do**
14: **if** $v.color =$ BLANCO **then**
15: $v.color \leftarrow$ GRIS
16: $v.d \leftarrow u.d + 1$
17: $v.\pi \leftarrow u$
18: Encolar($Q$, $v$)
19: **end if**
20: **end for**
21: $u.color \leftarrow$ NEGRO
22: **end while**




$Q$ | $s$ |
0

- Salida:

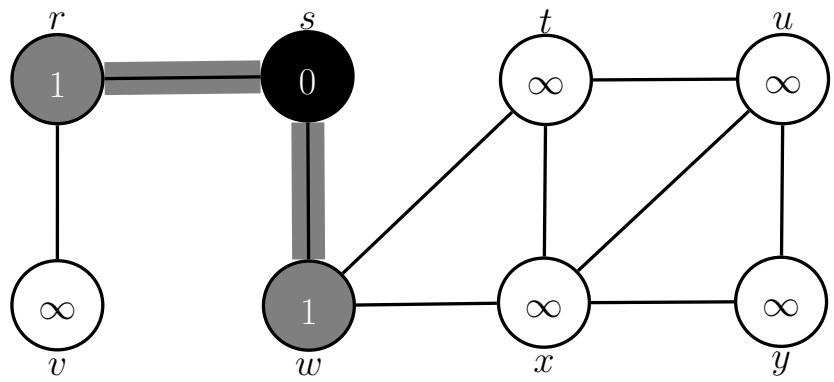
# Recorrido de Grafos

## Búsqueda primero en anchura

**Algorithm** BFS

**Require:** $G$: Grafo, $s$: vértice inicial.

1: **for** cada vértice $u \in G.V - \{s\}$ **do**
2: $u.color \leftarrow$ BLANCO
3: $u.d \leftarrow \infty$
4: $u.\pi \leftarrow NULL$
5: **end for**
6: $s.color \leftarrow$ GRIS
7: $s.d \leftarrow 0$
8: $s.\pi \leftarrow NULL$
9: $Q \leftarrow \emptyset$
10: Encolar($Q$, $s$)
11: **while** $Q \neq \emptyset$ **do**
12: $u \leftarrow$ Desencolar($Q$)
13: **for** cada $v \in G.Adyacente(u)$ **do**
14: **if** $v.color =$ BLANCO **then**
15: $v.color \leftarrow$ GRIS
16: $v.d \leftarrow u.d + 1$
17: $v.\pi \leftarrow u$
18: Encolar($Q$, $v$)
19: **end if**
20: **end for**
21: $u.color \leftarrow$ NEGRO
22: **end while**




$Q$

| $w$ | $r$ |
|---|---|
| 1 | 1 |

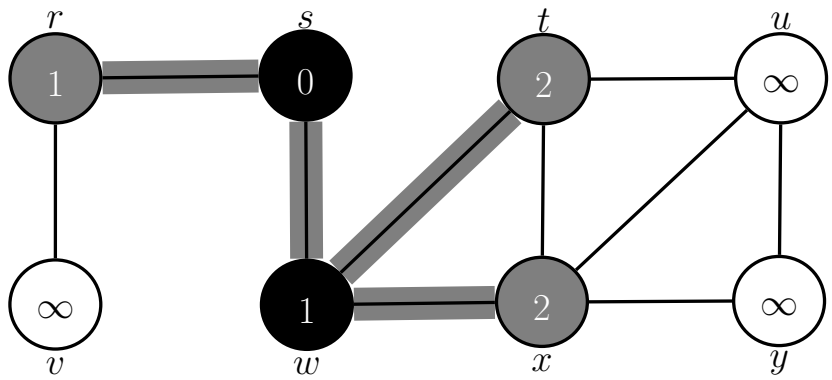
- Salida: $s$.

# Recorrido de Grafos

## Búsqueda primero en anchura

**Algorithm** BFS

**Require:** $G$: Grafo, $s$: vértice inicial.
1: **for** cada vértice $u \in G.V - \{s\}$ **do**
2: $u.color \leftarrow$ BLANCO
3: $u.d \leftarrow \infty$
4: $u.\pi \leftarrow NULL$
5: **end for**
6: $s.color \leftarrow$ GRIS
7: $s.d \leftarrow 0$
8: $s.\pi \leftarrow NULL$
9: $Q \leftarrow \emptyset$
10: Encolar($Q$, $s$)
11: **while** $Q \neq \emptyset$ **do**
12: $u \leftarrow$ Desencolar($Q$)
13: **for** cada $v \in G.Adyacente(u)$ **do**
14: **if** $v.color =$ BLANCO **then**
15: $v.color \leftarrow$ GRIS
16: $v.d \leftarrow u.d + 1$
17: $v.\pi \leftarrow u$
18: Encolar($Q$, $v$)
19: **end if**
20: **end for**
21: $u.color \leftarrow$ NEGRO
22: **end while**




| $Q$ | $r$ | $t$ | $x$ |
|---|---|---|---|
| | 1 | 2 | 2 |

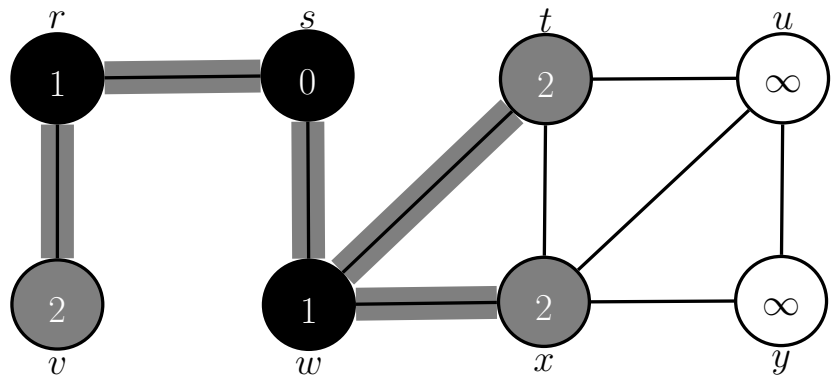
- Salida: $s, w$.

# Recorrido de Grafos

## Búsqueda primero en anchura

**Algorithm** BFS

**Require:** $G$: Grafo, $s$: vértice inicial.
1: **for** cada vértice $u \in G.V - \{s\}$ **do**
2: $u.color \leftarrow$ BLANCO
3: $u.d \leftarrow \infty$
4: $u.\pi \leftarrow NULL$
5: **end for**
6: $s.color \leftarrow$ GRIS
7: $s.d \leftarrow 0$
8: $s.\pi \leftarrow NULL$
9: $Q \leftarrow \emptyset$
10: Encolar($Q$, $s$)
11: **while** $Q \neq \emptyset$ **do**
12: $u \leftarrow$ Desencolar($Q$)
13: **for** cada $v \in G.Adyacente(u)$ **do**
14: **if** $v.color =$ BLANCO **then**
15: $v.color \leftarrow$ GRIS
16: $v.d \leftarrow u.d + 1$
17: $v.\pi \leftarrow u$
18: Encolar($Q$, $v$)
19: **end if**
20: **end for**
21: $u.color \leftarrow$ NEGRO
22: **end while**




| $Q$ | $t$ | $x$ | $v$ |
|---|---|---|---|
| | 2 | 2 | 2 |

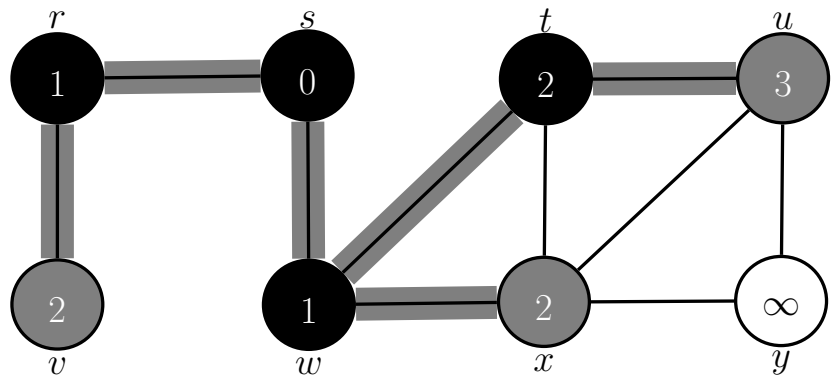
- Salida: $s, w, r$.

# Recorrido de Grafos

## Búsqueda primero en anchura

**Algorithm** BFS

**Require:** $G$: Grafo, $s$: vértice inicial.
1: **for** cada vértice $u \in G.V - \{s\}$ **do**
2: $u.color \leftarrow$ BLANCO
3: $u.d \leftarrow \infty$
4: $u.\pi \leftarrow NULL$
5: **end for**
6: $s.color \leftarrow$ GRIS
7: $s.d \leftarrow 0$
8: $s.\pi \leftarrow NULL$
9: $Q \leftarrow \emptyset$
10: Encolar($Q$, $s$)
11: **while** $Q \neq \emptyset$ **do**
12: $u \leftarrow$ Desencolar($Q$)
13: **for** cada $v \in G.Adyacente(u)$ **do**
14: **if** $v.color =$ BLANCO **then**
15: $v.color \leftarrow$ GRIS
16: $v.d \leftarrow u.d + 1$
17: $v.\pi \leftarrow u$
18: Encolar($Q$, $v$)
19: **end if**
20: **end for**
21: $u.color \leftarrow$ NEGRO
22: **end while**




| $Q$ | $x$ | $v$ | $u$ |
|---|---|---|---|
| | 2 | 2 | 3 |

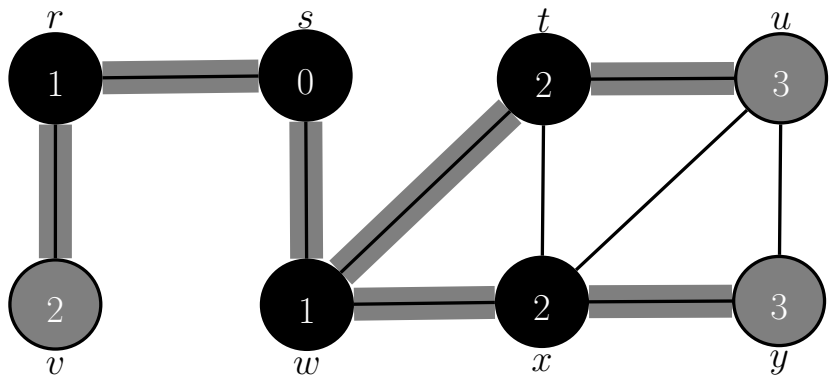
- Salida: $s, w, r, t$.

# Recorrido de Grafos

## Búsqueda primero en anchura

**Algorithm** BFS

**Require:** $G$: Grafo, $s$: vértice inicial.
1: **for** cada vértice $u \in G.V - \{s\}$ **do**
2: $u.color \leftarrow$ BLANCO
3: $u.d \leftarrow \infty$
4: $u.\pi \leftarrow NULL$
5: **end for**
6: $s.color \leftarrow$ GRIS
7: $s.d \leftarrow 0$
8: $s.\pi \leftarrow NULL$
9: $Q \leftarrow \emptyset$
10: Encolar($Q$, $s$)
11: **while** $Q \neq \emptyset$ **do**
12: $u \leftarrow$ Desencolar($Q$)
13: **for** cada $v \in G.Adyacente(u)$ **do**
14: **if** $v.color =$ BLANCO **then**
15: $v.color \leftarrow$ GRIS
16: $v.d \leftarrow u.d + 1$
17: $v.\pi \leftarrow u$
18: Encolar($Q$, $v$)
19: **end if**
20: **end for**
21: $u.color \leftarrow$ NEGRO
22: **end while**




| $Q$ | $v$ | $u$ | $y$ |
|---|---|---|---|
| | 2 | 3 | 3 |

- Salida: $s, w, r, t, x$.

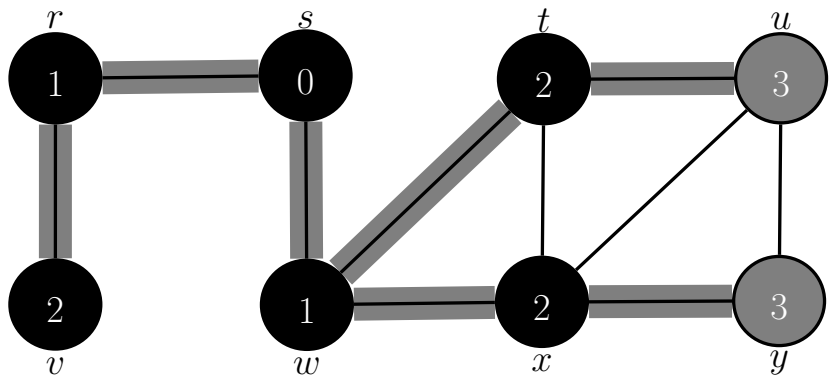
# Recorrido de Grafos

## Búsqueda primero en anchura

**Algorithm** BFS

**Require:** $G$: Grafo, $s$: vértice inicial.

1: **for** cada vértice $u \in G.V - \{s\}$ **do**
2: $u.color \leftarrow$ BLANCO
3: $u.d \leftarrow \infty$
4: $u.\pi \leftarrow NULL$
5: **end for**
6: $s.color \leftarrow$ GRIS
7: $s.d \leftarrow 0$
8: $s.\pi \leftarrow NULL$
9: $Q \leftarrow \emptyset$
10: Encolar($Q$, $s$)
11: **while** $Q \neq \emptyset$ **do**
12: $u \leftarrow$ Desencolar($Q$)
13: **for** cada $v \in G.Adyacente(u)$ **do**
14: **if** $v.color =$ BLANCO **then**
15: $v.color \leftarrow$ GRIS
16: $v.d \leftarrow u.d + 1$
17: $v.\pi \leftarrow u$
18: Encolar($Q$, $v$)
19: **end if**
20: **end for**
21: $u.color \leftarrow$ NEGRO
22: **end while**




$Q$

| $u$ | $y$ |
|---|---|
| 3 | 3 |

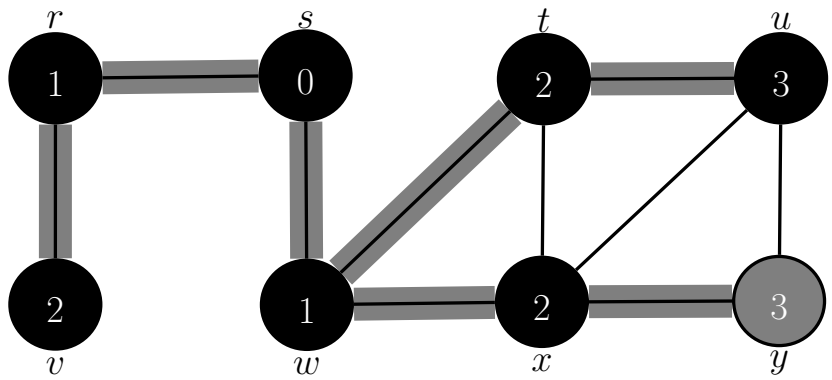
- Salida: $s, w, r, t, x, v$.

# Recorrido de Grafos

## Búsqueda primero en anchura

**Algorithm** BFS

**Require:** $G$: Grafo, $s$: vértice inicial.
1: **for** cada vértice $u \in G.V - \{s\}$ **do**
2: $u.color \leftarrow$ BLANCO
3: $u.d \leftarrow \infty$
4: $u.\pi \leftarrow NULL$
5: **end for**
6: $s.color \leftarrow$ GRIS
7: $s.d \leftarrow 0$
8: $s.\pi \leftarrow NULL$
9: $Q \leftarrow \emptyset$
10: Encolar($Q$, $s$)
11: **while** $Q \neq \emptyset$ **do**
12: $u \leftarrow$ Desencolar($Q$)
13: **for** cada $v \in G.Adyacente(u)$ **do**
14: **if** $v.color =$ BLANCO **then**
15: $v.color \leftarrow$ GRIS
16: $v.d \leftarrow u.d + 1$
17: $v.\pi \leftarrow u$
18: Encolar($Q$, $v$)
19: **end if**
20: **end for**
21: $u.color \leftarrow$ NEGRO
22: **end while**




$Q$ | $y$ | 3

- Salida: $s, w, r, t, x, v, u$.

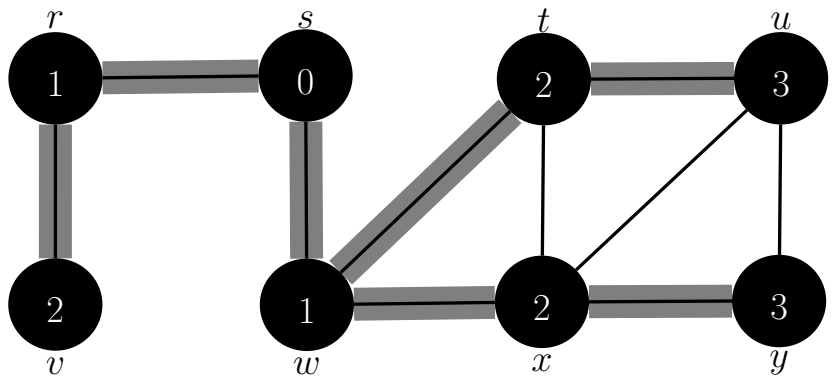
# Recorrido de Grafos

## Búsqueda primero en anchura

**Algorithm** BFS

**Require:** $G$: Grafo, $s$: vértice inicial.

1: **for** cada vértice $u \in G.V - \{s\}$ **do**
2: $u.color \leftarrow$ BLANCO
3: $u.d \leftarrow \infty$
4: $u.\pi \leftarrow NULL$
5: **end for**
6: $s.color \leftarrow$ GRIS
7: $s.d \leftarrow 0$
8: $s.\pi \leftarrow NULL$
9: $Q \leftarrow \emptyset$
10: Encolar($Q$, $s$)
11: **while** $Q \neq \emptyset$ **do**
12: $u \leftarrow$ Desencolar($Q$)
13: **for** cada $v \in G.Adyacente(u)$ **do**
14: **if** $v.color =$ BLANCO **then**
15: $v.color \leftarrow$ GRIS
16: $v.d \leftarrow u.d + 1$
17: $v.\pi \leftarrow u$
18: Encolar($Q$, $v$)
19: **end if**
20: **end for**
21: $u.color \leftarrow$ NEGRO
22: **end while**




$Q$ [ — ]

- Salida: $s, w, r, t, x, v, u, y$.

# Recorrido de Grafos

## Búsqueda primero en anchura

**Algorithm** BFS

**Require:** $G$: Grafo, $s$: vértice inicial.

```
for cada vértice u ∈ G.V − {s} do
    u.color ← BLANCO
    u.d ← ∞
    u.π ← NULL
end for
s.color ← GRIS
s.d ← 0
s.π ← NULL
Q ← ∅
Encolar(Q, s)
while Q ≠ ∅ do
    u ← Desencolar(Q)
    for cada v ∈ G.Adyacente(u) do
        if v.color = BLANCO then
            v.color ← GRIS
            v.d ← u.d + 1
            v.π ← u
            Encolar(Q, v)
        end if
    end for
    u.color ← NEGRO
end while
```

- Salida: $s$, $w$, $r$, $t$, $x$, $v$, $u$, $y$.
- $O(|V| + |E|)$